STM32F373xx Errata sheet

STM32F373xx device limitations

Silicon identification

This errata sheet applies to revision B of the STMicroelectronics STM32F373xx products. These families feature an ARM® 32-bit Cortex®-M4 core, for which an errata notice is also available (see *Section 1* for details).

Section 2 gives a detailed description of the product silicon limitations.

The full list of part numbers is shown in *Table 2*. The products are identifiable as shown in *Table 1*:

- by the revision code marked below the order code on the device package
- by the last three digits of the Internal order code printed on the box label

Table 1. Device identification(1)

Order code	Revision code(2) marked on device
STM32F373xx	“B”

1. The REV_ID bits in the DBGMCU_IDCODE register show the revision code of the device (see the RM0313 reference manual for details on how to find the revision code).
2. Refer to the device datasheet for details on how to identify the revision code on the different packages.

Table 2. Device summary

Reference	Part number
STM32F373xx	STM32F373C8, STM32F373R8, STM32F373V8, STM32F373CB, STM32F373RB, STM32F373VB, STM32F373CC, STM32F373RC, STM32F373VC

Contents

1 ARM® 32-bit Cortex®-M4 limitations 5

1.1 Cortex®-M4 interrupted loads to stack pointer can cause erroneous behavior 5

1.2 VDIV or VSQRT instructions might not complete correctly when very short ISRs are used 6

2 STM32F373xx silicon limitations 7

2.1 System limitations 9

2.1.1 Wakeup sequence from Standby mode when using more than one wakeup source 9

2.1.2 Delay after an RCC peripheral clock enabling 9

2.1.3 Full JTAG configuration without NJTRST pin cannot be used 10

2.2 CRC limitation 10

2.2.1 CRC corrupted when even polynomial is used 10

2.3 USART peripheral limitations 10

2.3.1 Communication parameters reprogramming after ATR in Smartcard mode when SCLK is used to clock the card 10

2.3.2 Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR 10

2.3.3 Start bit detected too soon when sampling for NACK signal from the smartcard 11

2.3.4 Break request can prevent the Transmission Complete flag (TC) from being set 11

2.3.5 nRTS is active while RE or UE = 0 11

2.4 SDADC peripheral limitation 12

2.4.1 SDADC incorrect gain amplification in single-ended zero reference mode for 16x and 32x gains 12

2.5 SPI/I2S peripheral limitations 12

2.5.1 Packing mode limitation at reception 12

2.5.2 SPI CRC may be corrupted when a peripheral connected to the same DMA channel of the SPI is under DMA transaction near the end of transfer or end of transfer ‘-1’ 13

2.5.3 In I2S slave mode, WS level must be set by the external master when enabling the I2S 13

2.6 I²C peripheral limitations 14

2.6.1 10-bit Slave mode: wrong direction bit value after Read header reception 14

2.6.2 10-bit combined with 7-bit Slave mode: ADDCODE may indicate wrong slave address detection . . . 14
2.6.3 Wakeup frames may not wake up the MCU when STOP mode entry follows enabling I^2C . . . 15
2.6.4 Wrong behaviors in Stop mode when wakeup from Stop mode is disabled in I^2C . . . 15
2.6.5 Wakeup frame may not wakeup from STOP if $t_{HD(STA)}$ is close to startup time . . . 16
2.6.6 Wrong data sampling when data set-up time ($t_{SU;DAT}$) is smaller than one I2CCLK period . . . 17
2.7 GPIO peripheral limitation . . . 17
2.7.1 GPIOx locking mechanism not working properly for GPIOx_OTYPE register . . . 17
2.8 Comparator peripheral limitation . . . 18
2.8.1 VREFINT scaler startup time from power down parameter degradation 18
3 Revision history . . . 19

List of tables

Table 1. Device identification 1
Table 2. Device summary 1
Table 3. Cortex®-M4 core limitations and impact on microcontroller behavior 5
Table 4. Summary of silicon limitations 7
Table 5. Document revision history 19

1 ARM® 32-bit Cortex®-M4 limitations

An errata notice of the STM32F373xx core is available from the following web address: http://infocenter.arm.com.

All the described limitations are minor and related to the revision r0p1-v1 of the Cortex®-M4 core. *Table 3* summarizes these limitations and their implications on the behavior of STM32F373xx devices.

Table 3. Cortex®-M4 core limitations and impact on microcontroller behavior

ARM ID	ARM category	ARM summary of errata	Impact on STM32F373xx
752770	Cat B	Interrupted loads to SP can cause erroneous behavior	Minor
776924	Cat B	VDIV or VSQRT instructions might not complete correctly when very short ISRs are used	Minor

1.1 Cortex®-M4 interrupted loads to stack pointer can cause erroneous behavior

Description

An interrupt occurring during the data-phase of a single word load to the stack pointer (SP/R13) can cause an erroneous behavior of the device. In addition, returning from the interrupt results in the load instruction being executed an additional time.

For all the instructions performing an update of the base register, the base register is erroneously updated on each execution, resulting in the stack pointer being loaded from an incorrect memory location.

The instructions affected by this limitation are the following:

- LDR SP, [Rn],#imm
- LDR SP, [Rn,#imm]!
- LDR SP, [Rn,#imm]
- LDR SP, [Rn]
- LDR SP, [Rn,Rm]

Workaround

As of today, no compiler generates these particular instructions. This limitation can only occur with hand-written assembly code.

Both issues can be solved by replacing the direct load to the stack pointer by an intermediate load to a general-purpose register followed by a move to the stack pointer.

Example:

Replace LDR SP, [R0] by

LDR R2,[R0]

MOV SP,R2

1.2 VDIV or VSQRT instructions might not complete correctly when very short ISRs are used

Description

On Cortex®-M4 with FPU core, 14 cycles are required to execute a VDIV or VSQRT instruction.
This limitation is present when the following conditions are met:

- A VDIV or VSQRT is executed
- The destination register for VDIV or VSQRT is one of s0 - s15
- An interrupt occurs and is taken
- The ISR being executed does not contain a floating point instruction
- 14 cycles after the VDIV or VSQRT is executed, an interrupt return is executed

In this case, if there are only one or two instructions inside the interrupt service routine, then the VDIV or VQSRT instruction does not complete correctly and the register bank and FPSCR are not updated, meaning that these registers hold incorrect out-of-date data.

Workarounds

Two workarounds are applicable:

- Disable lazy context save of floating point state by clearing LSPEN to 0 (bit 30 of the FPCCR at address 0xE000EF34).
- Ensure that every ISR contains more than 2 instructions in addition to the exception return instruction.

2 STM32F373xx silicon limitations

Table 4 gives quick references to all documented limitations.

The legend for *Table 4* is as follows:

A = workaround available,

N = no workaround available,

P = partial workaround available,

'-' and grayed = fixed.

Table 4. Summary of silicon limitations

Links to silicon limitations		Revision B
Section 2.1: System limitations	*Section 2.1.1: Wakeup sequence from Standby mode when using more than one wakeup source*	A
	Section 2.1.2: Delay after an RCC peripheral clock enabling	A
	Section 2.1.3: Full JTAG configuration without NJTRST pin cannot be used	A
Section 2.2: CRC limitation	*Section 2.2.1: CRC corrupted when even polynomial is used*	A
Section 2.3: USART peripheral limitations	*Section 2.3.1: Communication parameters reprogramming after ATR in Smartcard mode when SCLK is used to clock the card*	A
	Section 2.3.2: Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR	A
	Section 2.3.3: Start bit detected too soon when sampling for NACK signal from the smartcard	N
	Section 2.3.4: Break request can prevent the Transmission Complete flag (TC) from being set	A
	Section 2.3.5: nRTS is active while RE or UE = 0	A
Section 2.4: SDADC peripheral limitation	*Section 2.4.1: SDADC incorrect gain amplification in single-ended zero reference mode for 16x and 32x gains*	A
Section 2.5: SPI/I2S peripheral limitations	*Section 2.5.1: Packing mode limitation at reception*	N
	Section 2.5.2: SPI CRC may be corrupted when a peripheral connected to the same DMA channel of the SPI is under DMA transaction near the end of transfer or end of transfer '-1'	A
	Section 2.5.3: In I2S slave mode, WS level must be set by the external master when enabling the I2S	A

Table 4. Summary of silicon limitations (continued)

Links to silicon limitations		Revision B
Section 2.6: I2C peripheral limitations	*Section 2.6.1: 10-bit Slave mode: wrong direction bit value after Read header reception*	A
	Section 2.6.2: 10-bit combined with 7-bit Slave mode: ADDCODE may indicate wrong slave address detection	N
	Section 2.6.3: Wakeup frames may not wake up the MCU when STOP mode entry follows enabling I2C	A
	Section 2.6.4: Wrong behaviors in Stop mode when wakeup from Stop mode is disabled in I2C	A
	Section 2.6.5: Wakeup frame may not wakeup from STOP if tHD(STA) is close to startup time	N
	Section 2.6.6: Wrong data sampling when data set-up time (tSU;DAT) is smaller than one I2CCLK period	A
Section 2.7: GPIO peripheral limitation	*Section 2.7.1: GPIOx locking mechanism not working properly for GPIOx_OTYPE register*	A
Section 2.8: Comparator peripheral limitation	*Section 2.8.1: VREFINT scaler startup time from power down parameter degradation*	N

2.1 System limitations

2.1.1 Wakeup sequence from Standby mode when using more than one wakeup source

Description

The various wakeup sources are logically OR-ed in front of the rising-edge detector which generates the wakeup flag (WUF). The WUF needs to be cleared prior to Standby mode entry, otherwise the MCU wakes up immediately.

If one of the configured wakeup sources is kept high during the clearing of the WUF (by setting the CWUF bit), it may mask further wakeup events on the input of the edge detector. As a consequence, the MCU might not be able to wake up from Standby mode.

Workaround

To avoid this problem, the following sequence should be applied before entering Standby mode:

- Disable all used wakeup sources,
- Clear all related wakeup flags,
- Re-enable all used wakeup sources,
- Enter Standby mode

Note: Be aware that, when applying this workaround, if one of the wakeup sources is still kept high, the MCU will enter Standby mode but then it wakes up immediately generating a power reset.

2.1.2 Delay after an RCC peripheral clock enabling

Description

A delay between an RCC peripheral clock enable and the effective peripheral enabling should be taken into account in order to manage the peripheral read/write from/to registers. This delay depends on the peripheral mapping.

If peripheral is mapped on AHB: the delay is 2 AHB clock cycles after the clock enable bit is set on the hardware register.

If peripheral is mapped on APB: the delay is 2 APB clock cycles after the clock enable bit is set on the hardware register.

Workarounds

1. Enable the peripheral clock sometimes before the peripheral read/write register is required.
2. For AHB peripheral, insert two dummy read to the peripheral register.
3. For APB peripheral, insert a dummy read to the peripheral register.

2.1.3 Full JTAG configuration without NJTRST pin cannot be used

Description

When using the JTAG debug port in debug mode, the connection with the debugger is lost if the NJTRST pin (PB4) is used as a GPIO. Only the 4-wire JTAG port configuration is impacted.

Workaround

Use the SWD debug port instead of the full 4-wire JTAG port.

2.2 CRC limitation

2.2.1 CRC corrupted when even polynomial is used

Description

When selecting an *even* polynomial, the CRC is corrupted.

Workaround

Use an *odd* polynomial.

2.3 USART peripheral limitations

2.3.1 Communication parameters reprogramming after ATR in Smartcard mode when SCLK is used to clock the card

Description

If the USART is used in Smartcard mode and the card cannot use the default communication parameters after Answer To Reset and doesn't support clock stop, it is not possible to use SCLK to clock the card. This is due to the fact that the USART and its clock output must be disabled while reprogramming some of the parameters.

Workaround

Use another clock source to clock the card (e.g. a timer output programmed to the desired clock frequency).

2.3.2 Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR

Description

If the USART clock source is slow (for example LSE) and TE bit is cleared immediately after the last write to TDR, the last byte will probably not be transmitted.

Workarounds

1. Wait until TXE flag is set before clearing TE bit.

2. Wait until TC flag is set before clearing TE bit.

2.3.3 Start bit detected too soon when sampling for NACK signal from the smartcard

Description

In the ISO7816, when a character parity error is incorrect, the smartcard receiver shall transmit a NACK error signal at (10.5 +/- 0.2) etu after the character START bit falling edge. In this case, the USART transmitter should be able to detect correctly the NACK signal by sampling at (11.0 +/-0.2) etu after the character START bit falling edge.

The USART peripheral used in smartcard mode doesn't respect the (11 +/-0.2) etu timing, and when the NACK falling edge arrives at 10.68 etu or later, the USART might misinterpret this transition as a START bit even if the NACK is correctly detected.

Workaround

None.

2.3.4 Break request can prevent the Transmission Complete flag (TC) from being set

Description

After the end of transmission of a data (D1), the Transmission Complete (TC) flag will not be set in the following conditions:

- CTS hardware flow control is enabled.
- D1 is being transmitted.
- A break transfer is requested before the end of D1 transfer.
- nCTS is de-asserted before the end of transfer of D1.

Workaround

If the application needs to detect the end of transfer of the data, the break request should be done after making sure that the TC flag is set.

2.3.5 nRTS is active while RE or UE = 0

Description

The nRTS line is driven low as soon as RTSE bit is set even if the USART is disabled (UE = 0) or the receiver is disabled (RE = 0) i.e. not ready to receive data.

Workaround

Configure the I/O used for nRTS as alternate function after setting the UE and RE bits.

2.4 SDADC peripheral limitation

2.4.1 SDADC incorrect gain amplification in single-ended zero reference mode for 16x and 32x gains

Description

When using the Single-ended mode (zero reference) for 16x and 32x gains, the device subtracts incorrectly the analog offset when performing the digital multiplication (for 2x and 4x respectively while using analog gain 8x), and the results obtained in Single-ended mode (zero reference) for gains 16x and 32x will be incorrect.

Workaround

Use Single-ended offset mode instead of Single-ended mode (zero reference) for gains 16x and 32x. This mode also features better dynamic characteristics for these gains.

2.5 SPI/I2S peripheral limitations

2.5.1 Packing mode limitation at reception

Description

When the SPI is configured in the short data frame mode, the packing mode on the reception side may not be usable. Using this feature may generate a wrong RXNE event to an Interrupt or DMA request and so the software may read back inconsistent data with FIFO pointers misalignment on the reception FIFO.

The worst case is the Slave mode if the external master is running in continuous mode without clock interruption between two data transfers.

In full duplex Master mode, it runs correctly if the SPI is working in non-continuous mode, meaning that the SPI is transferring two data, then stopping the data transmission until the two data received are read back before sending the next two data.

Conditions to see this limitation:

- Packing mode is used
- SPI master (in continuous mode) or Slave (worst case)
- Full duplex or receiver mode

If the packing mode is used in reception mode, the FIFO reception threshold has to be set to 16 bit. Under those setting and conditions, when a read operation (half-word to read two data in one APB access) takes place while the FIFO level is equal to 3/4 (new data came before the two first ones are read), the 16-bit read decreases the FIFO level to 1/4. The RXNE flag is not de-asserted (clear condition on FIFO empty event) and a new request is present to read back two data although the FIFO contains only one data. Read and write pointers in the FIFO become misaligned and the data is corrupted.

The packing mode in reception has to be discarded when the conditions described above are met. It means that the reception FIFO requests that the data is read back until the FIFO content is empty. It also means that for short data frame (the worst case being the 4-bit data size), if the software or the DMA is not able to manage the high data rate when the SPI is running full speed, an Overrun condition may occur at regular intervals.

Workaround

There is no workaround.

The only way to avoid this overrun condition would be to slow down the SPI communication clock frequency in order to let time to the DMA (best case) to read back data without any FIFO full condition.

2.5.2 SPI CRC may be corrupted when a peripheral connected to the same DMA channel of the SPI is under DMA transaction near the end of transfer or end of transfer '-1'

Description

SPI CRC may be corrupted when a peripheral connected to the same DMA channel of the SPI is under DMA transaction near the end of transfer or end of transfer '*-1'.*

In the following conditions:

- SPI is slave or master,
- Full duplex or simplex mode is used,
- CRC feature is enabled,
- SPI is configured to manage data transfers by software (interrupt or polling),
- a peripheral, mapped on the same DMA channel as the SPI, is doing DMA transfers,

the CRC may be frozen before the CRCNEXT bit is written, resulting in a CRC error.

Workaround

If the application allows it, use the DMA for SPI transfers.

2.5.3 In I2S slave mode, WS level must be set by the external master when enabling the I2S

Description

In slave mode the WS signal level is used only to start the communication. If the I2S (in slave mode) is enabled while the master is already sending the clock and the WS signal level is low (for I2S protocol) or high (for the LSB or MSB-justified mode), the slave starts communicating data immediately. In this case the master and slave will be desynchronized throughout the whole communication.

Workaround

The I2S peripheral must be enabled when the external master sets the WS line at:

- High level when the I2S protocol is selected.
- Low level when the LSB or MSB-justified mode is selected.

2.6 I²C peripheral limitations

2.6.1 10-bit Slave mode: wrong direction bit value after Read header reception

Description

Under specific conditions, the transfer direction bit DIR (bit 16 of status register I2C_ISR) is low instead of high after reception of the 10-bit addressing Read header. Nevertheless, the I²C operates correctly in Slave transmission mode, and data can be sent using the TXIS flag.

To see the limitation, all the following conditions have to be fulfilled:

- I²C has to be configured in 10-bit addressing mode (OA1MODE is set in the I2C_OAR1 register).
- The high LSBs of the I²C slave address are equal to the 10-bit addressing Read header value (i.e. OA1[7:3] = 11110, OA1[2] = OA1[9], OA1[1] = OA1[8] and OA1[0] = 1 in the I2C_OAR1 register).
- The I²C receives the 10-bit addressing Read header (0x 1111 0XX1) after the repeated start condition to enter Slave transmission mode.

As a result, the DIR bit is incorrect in Slave mode under specific conditions.

Workaround

If possible, do not use these four values as 10-bit addresses in Slave mode:

- OA1[9:0] = 0011110001
- OA1[9:0] = 0111110011
- OA1[9:0] = 1011110101
- OA1[9:0] = 1111110111

If one of these addresses is the I²C slave address, the DIR bit must not be used in the FW.

2.6.2 10-bit combined with 7-bit Slave mode: ADDCODE may indicate wrong slave address detection

Description

Under specific conditions, the ADDCODE (Address match code) in the I2C_ISR register indicates a wrong slave address.

To see the limitation, all the following conditions have to be fulfilled:

- The I²C slave address OA1 is enabled and configured in 10-bit mode (OA1EN=1 and OA1MODE=1)
- Another 7-bit slave address is enabled and the bits 1 to 7 of the 10-bit slave address OA1 are equal to the 7-bit slave address, i.e. one of the configurations below is set:
  - OA2EN=1 and OA2MSK = 0 and OA1[7:1] = OA2[7:1]
  - OA2EN=1 and OA2MSK = 1 and OA1[7:2] = OA2[7:2]
  - OA2EN=1 and OA2MSK = 2 and OA1[7:3] = OA2[7:3]
  - OA2EN=1 and OA2MSK = 3 and OA1[7:4] = OA2[7:4]
  - OA2EN=1 and OA2MSK = 4 and OA1[7:5] = OA2[7:5]
  - OA2EN=1 and OA2MSK = 5 and OA1[7:6] = OA2[7:6]
  - OA2EN=1 and OA2MSK = 6 and OA1[7] = OA2[7]
  - OA2EN=1 and OA2MSK = 7
  - GCEN=1 and OA1[7:1] = 0b0000000
  - ALERTEN=1 and OA1[7:1] = 0b0001100
  - SMBDEN=1 and OA1[7:1] = 0b1100001
  - SMBHEN=1 and OA1[7:1] = 0b0001000
- The master starts a transfer addressed to the 10-bit slave address OA1.

As a result, after the address reception, the ADDCODE value is OA1[7:1] equal to the 7-bit slave address, instead of 0b11110 & OA1[9:8].

Workaround

None. If several slave address are enabled, mixing 10-bit and 7-bit addresses, the 10-bit slave address OA1 [7:1] must not be equal to the 7-bit slave address.

2.6.3 Wakeup frames may not wake up the MCU when STOP mode entry follows enabling I²C

Description

If the I²C is enabled (PE = 1) and wakeup from STOP is enabled in I²C (WUPEN=1) while a transfer occurs on the I²C bus and STOP mode is entered during the same transfer while SCL=0, the I²C is not able to detect the first following START condition. This means that if the I²C is addressed, it will not wake up the MCU and this address is not acknowledged.

Workaround

After enabling the I²C (PE is set to 1), wait for a temporization before entering STOP mode, to ensure that the eventual on-going frame is finished.

2.6.4 Wrong behaviors in Stop mode when wakeup from Stop mode is disabled in I²C

Description

When wakeup from Stop mode is disabled in I²C (WUPEN = 0) and the MCU enters Stop mode while a transfer is on going on the bus, some wrong behaviors may happen:

1. BUSY flag can be wrongly set when the MCU exits Stop mode. This prevents from initiating a transfer in master mode, as the START condition cannot be sent when BUSY is set.
2. If clock stretching is enabled (NOSTRETCH = 0), the I^2C clock SCL may be stretched low by the I^2C as long as the MCU is in Stop mode. This limitation may occur when the Stop mode is entered during the address phase of a transfer on the I^2C bus while SCL = 0. Therefore the transfer may be stalled as long as the MCU is in Stop mode. The probability of the occurrence depends also on the timings configuration, the peripheral clock frequency and the I^2C bus frequency.

These behaviors can occur in Slave mode and in Master mode in a multi-master topology.

Workaround

Disable the I^2C (PE=0) before entering Stop mode and re-enable it in Run mode.

2.6.5 Wakeup frame may not wakeup from STOP if $t_{HD(STA)}$ is close to startup time

Description

Under specific conditions and if the START condition hold time $t_{HD(STA)}$ duration is very close to the HSI startup time duration, the I^2C is not able to detect the address match and to wake up the MCU from STOP. To see the limitation, one of the conditions listed below has to be met:

- Timeout detection is enabled (TIMOUTEN=1 or TEXTEN=1) and the frame before the wakeup frame is abnormally finished due to a I^2C Timeout detection (TIMOUT=1).
- The slave arbitration is lost during the frame before the wakeup frame (ARLO=1).
- The MCU enters STOP mode while another slave is addressed, after the address phase and before the STOP condition (BUSY=1).
- The MCU is in STOP mode and another slave is addressed before the I^2C is addressed.

Note: The last three conditions can occur only in a multi-slave network.

In STOP mode, the HSI is switched on by the I^2C when a START condition is detected (SDA falling edge while SCL is high). The HSI is used to receive the address. HSI is switched off after the address reception if received address is not the I^2C slave address. If one of the conditions above is met and if the SCL falling edge following the START condition occurs on the first cycle of the I2CCLK clock (HSI), the address reception is not correctly done and the address match wakeup interrupt is not generated.

Workaround

- None at MCU level. If the wakeup frame is not acknowledged by the I^2C and if the master can program the duration of the START hold time: the master should decrease or increase the START condition hold time for more than one HSI period and resend the wakeup frame.

2.6.6 Wrong data sampling when data set-up time ($t_{SU;DAT}$) is smaller than one I2CCLK period

Description

The I2C bus specification and user manual specifies a minimum data set-up time ($t_{SU;DAT}$) at:

- 250ns in Standard-mode.
- 100 ns in Fast-mode.
- 50 ns in Fast-mode Plus.

The I2C SDA line is not correctly sampled when $t_{SU;DAT}$ is smaller than one I2CCLK (I2C clock) period: the previous SDA value is sampled instead of the current one. This can result in a wrong slave address reception, a wrong received data byte, or a wrong received acknowledge bit.

Workaround

Increase the I2CCLK frequency to get I2CCLK period smaller than the transmitter minimum data set-up time. Or, if it is possible, increase the transmitter minimum data set-up time.

2.7 GPIO peripheral limitation

2.7.1 GPIOx locking mechanism not working properly for GPIOx_OTYPE register

Description

Locking of GPIOx_OTYPER[i] with i = 15 ..8 depends from setting of GPIOx_LCKR[i-8] and not from GPIOx_LCKR[i]. GPIOx_LCKR[i-8] is locking GPIOx_OTYPER[i] together with GPIOx_OTYPER[i-8]. It is not possible to lock GPIOx_OTYPER[i] with i = 15...8, without locking also GPIOx_OTYPER[i-8].

Workaround

The only way to lock GPIOxOYTYPE[i] with i=15..8 is to lock also GPIOxOYTYPE[i-8].

2.8 Comparator peripheral limitation

2.8.1 V_{REFINT} scaler startup time from power down parameter degradation

Description

The V_{REFINT} scaler is an embedded voltage follower providing the V_{REFINT} or its fractions (1/2, 1/4 or 3/4) to the comparator input.

The maximum V_{REFINT} scaler startup time, t_{S_SC}(max), is not as expected for the first activation of the V_{REFINT} scaler after powering on the device and it can be up to 1s (instead of 0.2ms) in worse case conditions. This maximum value depends mainly on the voltage and temperature, see the device datasheet for more details.

Workaround

None.

3 Revision history

Table 5. Document revision history

Date	Revision	Changes
06-Sep-2012	1	Initial release.
03-Oct-2013	2	Added: – *Section 2.1.1: Wakeup sequence from Standby mode when using more than one wakeup source* – *Section 2.2.1: CRC corrupted when even polynomial is used* – *Section 2.3.2: Last byte written in TDR might not be transmitted if TE is cleared just after writing in TDR* – *Section 2.5.2: SPI CRC may be corrupted when a peripheral connected to the same DMA channel of the SPI is under DMA transaction near the end of transfer or end of transfer '-1'* – *Section 2.6.5: Wakeup frame may not wakeup from STOP if tHD(STA) is close to startup time* Modified *Section 2.6.3: Wakeup frames may not wake up the MCU when STOP mode entry follows enabling I2C* and *Section 2.6.4: Wrong behaviors in Stop mode when wakeup from Stop mode is disabled in I2C*. Updated document disclaimer.
17-Feb-2014	3	Added: – *Section 2.5.3: In I2S slave mode, WS level must be set by the external master when enabling the I2S* – *Section 2.7.1: GPIOx locking mechanism not working properly for GPIOx_OTYPE register* Updated *Table 4: Summary of silicon limitations*. Added package UFBGA100 in *Appendix A: Revision code on device marking*. STM32F378xx (previously STM32F383xx) devices are now covered by a separate errata sheet.
02-Mar-2015	4	Added: - *Section 1.2: VDIV or VSQRT instructions might not complete correctly when very short ISRs are used*. - *Section 2.1.3: Full JTAG configuration without NJTRST pin cannot be used*. - *Section 2.6.6: Wrong data sampling when data set-up time (tSU;DAT) is smaller than one I2CCLK period*. - *Section 2.3.3: Start bit detected too soon when sampling for NACK signal from the smartcard*. - *Section 2.3.4: Break request can prevent the Transmission Complete flag (TC) from being set*. - *Section 2.3.5: nRTS is active while RE or UE = 0*. - *Section 2.1.2: Delay after an RCC peripheral clock enabling*. - *Section 2.8: Comparator peripheral limitation*.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2015 STMicroelectronics – All rights reserved